Vol.2014-SE-186 No.5
2014/11/13

大規模システム開発のテスト工程における ネットワークリプレイ適用の影響

臼井 崇文[1]　田中 修一[1]　秦野 康生[1]　森 拓郎[1]

概要：ソフトウェアやシステム開発の効率化・高品質化手法は各種あるが，大規模システム開発においては，さほど導入が進んでいない．そこで本報告では，現行開発手法からの変更点を少なくし，導入に際しての変更点を少なくした形で，リグレッションテストに対してネットワークリプレイを適用する手法を提案する．また，提案手法を結合テストに対して試行適用した結果，主目的であるテスト操作の自動化だけでなく，エビデンス保存の面においても高い有効性を発揮することが確認された．

Application of Network Replay Testing for Large-Scale System Development

Takafumi USUI[1]　Shuichi TANAKA[1]　Yasuo HATANO[1]　Takuro MORI[1]

Abstract: There are many techniques that efficient and high quality development of system or software, but are not used so much. So in this paper, we suggest that application of network replay techniques for regression test. Our application needs less changes compared with the current development. The result of applying, our application is also effective in storage of test evidence as well as test operation.

1. 研究背景

近年，ソフトウェアやシステム開発の効率化や高品質化を目指した各種技法が目覚ましい発展を遂げている [1]．中でもテストに関しては多様なテスト対象・テストフェーズ・テスト項目に向けて手法やツールが開発されており，いわゆるアジャイル開発においても重要な要素となっている [2][3]．

他方，エンタープライズシステム等の大規模システムの開発においては，これらの手法やツールの導入はさほど進んでいない．これら手法の多くは顧客との頻繁なやりとりや開発者間で密なレビュー・テストなどを行うことを前提としており，関係者や開発者の数も多くなる大規模システムの開発においては，実現が困難であるケースも多いためである [4]．

そこで本報告では，各種テスト手法の中でも現行手法からの変更点を比較的少なくして導入できるネットワークリプレイ手法を対象に，大規模システム開発へと適用する手法を提案する．次にこの手法を試行適用し，その結果をもとに各テスト工程が受ける影響や適用時に必要となる機能・支援など，大規模システム開発の現場へと各種技法の導入を進めていく際の課題とその解決案について検討した結果を報告する．

[1] (株) 日立製作所 横浜研究所
Hitachi Ltd. Yokohama Research Laboratory

2. ネットワークリプレイとその適用

2.1 ネットワークリプレイ

本報告で採用しているネットワークリプレイ手法はキャプチャリプレイ手法の一種であり，キャプチャして再利用する対象としてネットワークデータを選択した形態である．

ネットワークリプレイを含むキャプチャリプレイ手法は，テスト対象への操作をキャプチャするフェーズと，キャプチャした情報を利用してテスト対象への操作をリプレイすることでテストを行うフェーズとに分かれる．

まずキャプチャフェーズで，ユーザ操作を後から再生可能なように記録する．例えば Web アプリケーション向け

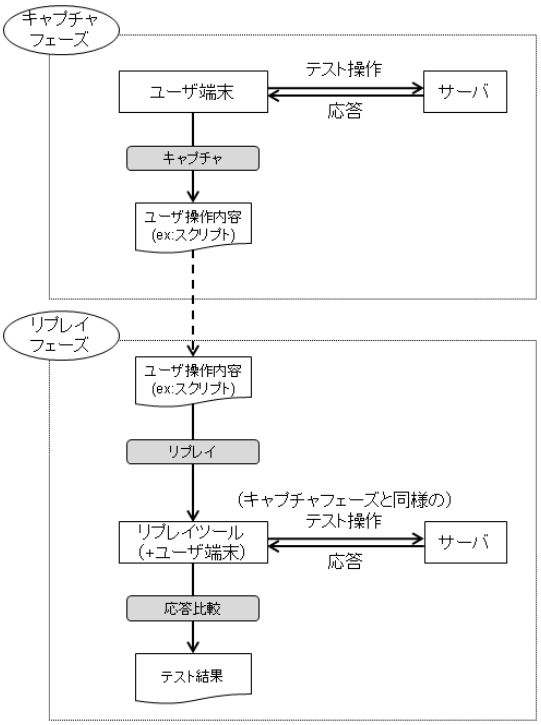

図 1 キャプチャリプレイ手法

にキャプチャリプレイを実装している Selenium[5] では，クリックや文字入力等のブラウザ上でのユーザ操作をスクリプトへと落とし込んで保存することが可能である．ここでネットワークリプレイ手法では，ユーザ端末とサーバ間でやり取りされているパケット等のネットワークデータを保存することになる．

次にリプレイフェーズで，記録しておいたユーザ操作に基づいて同様の操作の再実行を行う．先ほどと同様に Selenium を例にとると，Selenium はスクリプトに基づいてブラウザを操作し，キャプチャフェーズで行った操作を再実行する．また，サーバからの応答をあらかじめ用意した望ましい結果と比較することでテスト結果の確認を行い，同時にテストエビデンスの保存を行うことも可能である．ここでネットワークリプレイ手法では，保存しておいたネットワークデータを基にキャプチャフェーズと同様の通信を行うことになる．結果の確認においては，キャプチャフェーズで保存したサーバからの応答とリプレイフェーズで返されたサーバからの応答を比較することで，同様の操作を行った際に同様の結果が返ってきていることを確認することになる．

上記キャプチャリプレイ手法は，キャプチャしたユーザ操作をリプレイツールによって再実行させることを中心としている．つまり，手動で行っている操作をツールで代替させるものであり，何をどのようにテストするかといったテスト設計はキャプチャリプレイ適用前後で大きく変化しない．また，ユーザ操作のキャプチャもツール導入前のテスト環境をそのまま利用可能であるため，この点においても変化は少ない．したがって適用に際しての検討事項が少なくて済み，また，環境構築も容易であるため，適用が容易であるという特徴がある．

一方で，スクリプト等で保存されているユーザ操作を再実行する都合上，テスト対象の変化に弱いという欠点がある [6]．キャプチャしていない操作を必要とするテスト項目が追加された場合や，あるいは入力項目が増えたなどで追加の操作が必要になった場合などには，再キャプチャやスクリプトの修正などが必要となり，これらメンテナンス作業に対しても工数を割かなければならないことになる．

2.2 ネットワークリプレイの適用

前節で述べたネットワークリプレイ手法を，大規模システム開発へと適用することを考える．

適用の際に課題となるのは，以下の 2 点である．

- キャプチャフェーズが必要となる
- キャプチャ時と異なる操作をさせづらい

また，1 章で述べた通り，変更点を比較的少なくして適用できることを起点としてネットワークリプレイ手法の適用を検討しており，適用のための追加作業や環境構築等は可能な限り抑えることが望ましい．

以上の要件より筆者らは，リグレッションテストに対してネットワークリプレイ手法を適用することにした．適用イメージを図に示す．

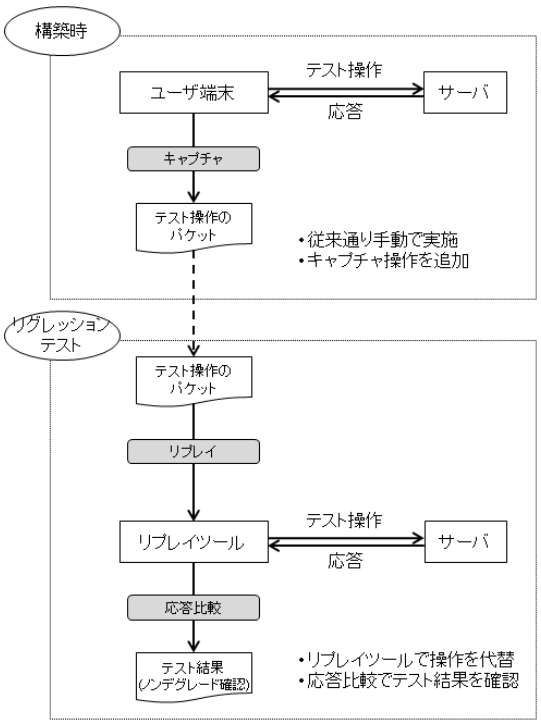

図 2 ネットワークリプレイ手法の適用イメージ

前述したように，ネットワークリプレイ手法を適用する際にはユーザ操作をキャプチャするフェーズが必要となる．今回の適用では，システム構築時の初回テストを行う

際にテスト操作のパケットをキャプチャすることとした．前節で述べたように，ネットワークリプレイ手法は収集したユーザ操作を再実行するため，操作が変更されるとキャプチャからやり直しとなってしまう．構築時の初回テストは特に多くの不具合が見つかり修正が発生するものと考えられるため，修正による操作変更やテスト内容の変更が度々生じる初回テスト時に用いるには，ネットワークリプレイは不向きである．そこで，構築時の初回テストは手動でテストを行うものとして，この時点でキャプチャを行って各種テストのユーザ操作を収集する．

その後，機能追加や改修が生じた場合にネットワークリプレイ手法を適用する．初回テスト時にキャプチャしておいたユーザ操作を再実行させることで，リグレッションテストを自動化する．中でも特に，追加や改修された機能に直接関わっていない機能について，バグが生じていないかどうかを確認するためにネットワークリプレイ手法を適用する．前述の通りネットワークリプレイ手法はキャプチャしたユーザ操作を再実行するものであり，キャプチャ時と異なる操作はさせづらい．データ駆動型テストのように入力項目とデータを外部に出し，例えば web フォームへの入力内容を種々に変更させることは可能であるが，入力項目が増えた場合や，あるいは操作手順が変わった場合には対応することが困難である．今回対象とする追加開発が生じた際のリグレッションテスト，特に，直接的に機能の追加や改修に関わっていない部分であればテスト操作も変更しなくて良いことが想定され，また，キャプチャ時のサーバからの応答とリグレッションテスト時のサーバからの応答を比較することでテスト結果の確認が容易に行えると想定されるためである．

上記のようにテスト操作と結果確認を自動化することで，多数の機能に対して多数のテスト項目を実施しなければならないリグレッションテストの工数削減・時間短縮が期待される．またそれに加えて，ネットワークリプレイ手法を適用する際にはツールを導入することになるが，このツールによってテスト結果のエビデンスを保存することも期待される．これは，現状ではテスト結果のエビデンス保存も手動で行うケースがあり，この点でも工数・時間がかかっているためである．

3. 適用試行

3.1 試行内容

前章で述べたネットワークリプレイ手法を適用し，適用前後でテスト操作，および結果確認とエビデンス保存で要した時間を計測した．適用対象は以下の通りである．

また，手順は以下の通りである．

(1) 対象システムに対して従来通りのテストを実施し，同時にキャプチャフェーズとしてパケットのキャプチャを行う．

表 1 適用対象

対象システム	エンタープライズシステム
テストフェーズ	結合テスト
テスト件数	39 件

(2) リプレイフェーズとして，キャプチャしたパケットを使用してリプレイツールでテスト操作を代替させる．

(3) キャプチャしたパケットとサーバからの応答パケットを突き合わせることで，テスト結果の確認とエビデンスの保存を行う．

なお，対象としたシステムが独自プロトコルを使用していたため，本適用に際してはネットワークリプレイツールを独自実装して使用している．

3.2 試行結果

試行結果を表 2 に示す．

表 2 試行結果

	従来手法	ネットワークリプレイ適用
テスト操作	35 分 09 秒	10 分 52 秒 *1
結果確認・エビデンス保存	235 分 51 秒	29 分 38 秒
合計	271 分	40 分 30 秒

なおここで，テスト操作は対象アプリケーションに対してテストシナリオ通りの操作を行う際に要した時間を指し，結果確認・エビデンス保存は操作結果の確認と，テスト結果を保存する際に要した時間を指す．

表 2 に示した通り，テスト操作と結果確認・エビデンス保存両者ともに，ネットワークリプレイ適用により作業時間短縮に効果があることが確認された．

テスト操作においては，従来手法では 60 秒/件程度必要であったシステムへの入力時間が，ネットワークリプレイツール適用後は 17 秒/件程度となっている．なお，今回実装したネットワークリプレイツールには，複数パケットを続けてリプレイする際にパケット間の時間間隔を詰められないという実装上の課題が残っており，この点を解決することで 2 秒/件程度までテスト操作時間を短縮できる見込みを得ている．

結果確認・エビデンス保存については，従来手法では 6 分/件程度必要であったところが，ネットワークリプレイツール適用により 46 秒/件程度となっている．適用後も結果確認でやや時間がかかっているものの，エビデンス保存が自動化された結果，必要時間が 1/8 程度となっていることが確認された．従来は 4 時間程度必要となっていた工程が 30 分程度まで短縮されており，特にこの点において高い効果を発揮することが確認された．

*1 ツール改善により，1 分 11 秒程度まで短縮見込み．

4. 各テスト工程が受ける影響

前章での試行結果より，ネットワークリプレイツール適用によってテスト操作，および結果確認・エビデンス保存の時間が短縮可能であることが確認された．本章では上記テスト操作の時間短縮やその他準備段階での検討事項なども含めて，各テスト工程が受ける影響について述べる．なおここでのテスト工程の分類は，[3] に従っている．

(1) テスト分析
テスト分析は，テスト対象のシステムやアプリケーションに対して，テストすべき内容を検討する工程を指す．
ネットワークリプレイ手法の適用によってテスト対象のシステムやアプリケーションが影響を受けるわけではなく，また，今回はリグレッションテストを対象としているため，本工程が受ける影響は小さいと考えられる．

(2) テスト設計
テスト設計は，テスト分析で検討された内容について，テストを行う具体的な方法を設定する工程を指す．
ネットワークリプレイ手法はその性質上，適用できないテスト項目がある．例えば画面レイアウトの確認はパケットからでは判断できないケースが多く，この点をテストしたい場合にはネットワークリプレイ手法は適用できないことになる．今回の試行においては事前に調査して適用可能なテスト項目にのみ適用しているが，実際には適用可能なテスト項目と適用不可能なテスト項目が混ざっており，各テスト項目に対して適用するか否かを判断しなければならない．あるいは適用可能なテスト項目であっても，テストデータの整備等の準備コストを考慮すると，適用しない方が結果が良くなるケースも想定される．したがって，テスト項目に対しての適用可否を判断する基準が明確であることが求められる．また，この適用可否の判断で従来よりも作業量が増えることになるため，理想的にはこの判断もツール化されていることが望ましい．

(3) テスト実装
テスト実装は，テスト設計で設定された方法について，実行するためのテストデータやツール等を準備する工程を指す．
ネットワークリプレイ手法の適用によりツールを使用することになるため，この準備において従来よりも作業量が増えてしまうことが想定される．今回想定している適用先はリグレッションテストの，特に非改修部分であるため，2 回目以降は初回の設定等が流用可能であり，作業量も比較的抑えられることが想定されるが，初回においては各種準備で作業量が大幅に増加することが想定され，適用の際のネックとなる．この点については，例えば設定ファイルの作成をある程度自動化するようなツールや，環境構築手順の整備 (あるいは，ツール自体が各種環境で動作可能なものにする)，ネットワークリプレイツールが GUI として操作方法が明確であること，などが求められる．

(4) テスト実行
テスト実行は，テスト実装で準備されたデータやツール等を使用して，実際にテストを行う工程を指す．
前章で述べた通りテスト実行においてはツール化の効果が発揮され，時間・工数が短縮されることになる．

(5) テスト結果管理
テスト結果管理は，テスト実行で行ったテストについて，意図通りに実施されていることや実行結果の確認・管理を行う工程を指す．
前章で述べた通りテスト結果の確認においてはツール化の効果が発揮され，時間・工数が短縮されることになる．一方で，テスト結果やテストエビデンスとして保存される情報が従来手法とは異なる場合が想定される．テスト結果については，ネットワークリプレイの場合はパケットの突き合わせ結果であり，これが従来手法と同様に問題ない結果であることを保証する必要がある．またテストエビデンスについても，システムの発注元に提出しなければならない場合もあるため，ツール適用後のエビデンスで十分であることを保証する仕組みが必要となる．加えて各種管理においても従来とは異なる情報を管理しなければならなくなるため，この点も整備されている必要がある．

(6) テストウェア管理
テストウェア管理は，これまでの工程で作成・準備されたテストケースやテストデータ，テスト用のツール等を管理する工程を指す．
ネットワークリプレイ適用によって増えたツールを管理する必要がある．これはツール本体だけではなく，適用の際の手順書なども含まれ，後から利用しやすい形態となっていることが望ましい．また，テストケースとテストデータ (ネットワークリプレイにおいてはキャプチャしたパケット) を関連付けて管理する必要がある．この点についても，管理の容易さと後日の再利用性・検索性を考慮した管理方法・ツールが求められる．

5. まとめと今後の課題

本報告では，エンタープライズシステム開発のリグレッションテストに対して，ネットワークリプレイ手法を適用する方法を提案した．また提案内容を結合テストにて試行適用し，特にエビデンス保存の面において高い効果を発揮することを確認した．また試行結果をもとに各テスト工程が受ける影響を検討し，特に事前準備の容易化と，エビデ

ンスの保存形式が変化しても従来手法同様の結果が得られていることが確認できる方法や，従来手法からは変化したエビデンスを含む管理手法などが適用を実現する上で重要であるとの結論を得た．

今後の課題は以下である．

(1) 結合テストにおける更なる評価

(2) 適用可能なテストの拡張

(1) に関しては，今回の試行適用では混ざって計測されている結果評価・エビデンス保存の時間を詳細に調査することや，事前準備の工数を測定するなどによるネットワークリプレイ手法適用による効果の更なる調査が必要である．また，今回はツールに慣れている報告者らがツールを使用してテストを行っているが，実際の開発者などツールに不慣れな者が行った場合に，どの程度の効果が出るかといった点も検討する必要がある．

(2) に関しては，今回は結合テストに対して適用を行っているが，例えばシステムテストなど別のテストに対して適用範囲を広げていくことが考えられる．これは特に顧客業務の流れを再現するシステムテストにおいてはテストシナリオ通りに長い操作を繰り返すことになるため，テストを自動化する手法が求められていることによる．

参考文献

[1] Boehm, B. and Turner, R.: Balancing Agility and Discipline: A Guide for the Perplexed, Addison-Wesley, Boston. (2004)
[2] Automated Testing Institute: Test Automation Body of Knowledge (TABOK), 入手先 ⟨http://www.automatedtestinginstitute.com⟩
[3] Association of Software Test Engineering (ASTER): テストツールまるわかりガイド（入門編），入手先 ⟨http://aster.or.jp/business/testtool_wg.html⟩
[4] 独立行政法人情報処理推進機構 (IPA): 非ウォーターフォール型開発に関する調査（調査報告書），入手先 ⟨http://www.ipa.go.jp/sec/softwareengineering/reports/20100330a.html⟩
[5] Selenium Projects: Selenium, 入手先 ⟨http://www.seleniumhq.org⟩
[6] Leotta, M., Clerissi, D., Ricca, F. and Tonella, P.: Capture-replay vs. programmable web testing: An empirical assessment during test case evolution, Reverse Engineering (WCRE), 2013 20th Working Conference (2013)